Boosted Web Named Entity Recognition via Tri-Training

CHIEN-LUNG CHOU, CHIA-HUI CHANG, and YA-YUN HUANG, National Central University

Named entity extraction is a fundamental task for many natural language processing applications on the web. Existing studies rely on annotated training data, which is quite expensive to obtain large datasets, limiting the effectiveness of recognition. In this research, we propose a semisupervised learning approach for web named entity recognition (NER) model construction via automatic labeling and tri-training. The former utilizes structured resources containing known named entities for automatic labeling, while the latter makes use of unlabeled examples to improve the extraction performance. Since this automatically labeled training data may contain noise, a self-testing procedure is used as a follow-up to remove low-confidence annotation and prepare higher-quality training data. Furthermore, we modify tri-training for sequence labeling and derive a proper initialization for large dataset training to improve entity recognition. Finally, we apply this semisupervised learning framework for person name recognition, business organization name recognition, and location name extraction. In the task of Chinese NER, an F-measure of 0.911, 0.849, and 0.845 can be achieved, for person, business organization, and location NER, respectively. The same framework is also applied for English and Japanese business organization name recognition and obtains models with performance of a 0.832 and 0.803 F-measure.

CCS Concepts: • **Computing methodologies** → **Artificial intelligence**; **Natural language processing**; **Information extraction**; **Semi-supervised learning settings;**

Additional Key Words and Phrases: Named entity recognition, tri-training for sequence labeling, tri-training initialization, and semisupervised learning

ACM Reference Format:
Chien-Lung Chou, Chia-Hui Chang, and Ya-Yun Huang. 2016. Boosted web named entity recognition via tri-training. ACM Trans. Asian Low-Resour. Lang. Inf. Process. 16, 2, Article 10 (October 2016), 23 pages.
DOI: http://dx.doi.org/10.1145/2963100

1. INTRODUCTION

The World Wide Web has now become the largest unstructured knowledge base in the world. Many applications such as question answering [Burger et al. 2001; Lin 2002], opinion mining [Hu and Liu 2004; Cambria et al. 2013], summarization [Yatsko et al. 2010], and point-of-interest (POI) search [Chang and Li 2010; Su 2012] have utilized web content to improve their performance and data quality. Meanwhile, with the flourishing of the social network, identifying important entities, topics, and events from online textual streams has gained increasing attention. For example, Vavliakis et al. [2013] proposed event detection from large timestamped web documents by integrating named entity recognition and topic modeling. Chuang et al. [2014] proposed an idea of address-bearing page crawling and business name recognition for POI extraction from the web. Gattani et al. [2013] introduced an industrial system that solves entity

This research was partially supported by the Industrial Technology Research Institute of Taiwan (ITRI) under grant B2-101052.
Authors' addresses: Department of Computer Science and Information Engineering, National Central University, Taiwan; emails: formatc.chou@gmail.com, chia@csie.ncu.edu.tw, a2425320032002@gmail.com.

DOI: http://dx.doi.org/10.1145/2963100

Table I. NER Packages Comparison

Category	NER Package	Precision	Recall	F-Measure
Chinese Person Name	FundanNLP	0.636	0.688	0.661
	Stanford NER	0.758	0.762	0.760
	Our System	0.936	0.887	0.911
Chinese Business Org. Name	FundanNLP	0.429	0.081	0.136
	Stanford NER	0.518	0.542	0.530
	Our System	0.825	0.875	0.849
Chinese Location Name	FundanNLP	0.353	0.377	0.365
	Stanford NER	0.215	0.188	0.201
	Our System	0.925	0.777	0.845

extraction, linking, classification, and tagging over social data. Among this research, named entity recognition (NER) is the most important step since it provides the key persons, locations, times, and even addresses and products that are mentioned on the web.

Like many research works, this task relies on annotated training examples that require large amounts of manual efforts. While human-labeled training examples have high quality, their cost is very high. Therefore, most tasks for NER are limited to several thousand sentences because of the high cost of labeling. For example, the English dataset for the CoNLL 2003 shared task consists of 14,987 sentences for four entity categories, PER, LOC, ORG, and MISC [Tjong Kim Sang and De Meulder 2003]. Thus, it is unclear whether sufficient data is provided for training or the learning algorithms have reached their capacity.

On the other hand, existing Chinese NER models such as the Stanford named entity recognizer [Finkel et al. 2005] are trained from a small set of news articles from the New China News Agency or People's Daily, which are more formal than the words used in blogs on the web. Therefore, the recognition performance is not as good as reported. For example, on a dataset of Google search results collected from 11,138 and 10,000 address queries, the Stanford NER on person name and business organization name recognition achieves only a 0.760 and 0.530 F-measure, which is far from satisfactory to be used for other web mining tasks (Table I). Similarly, FudanNLP [Qiu et al. 2013] gives only a 0.661 and 0.136 F-measure for person name and business organization name, respectively. As for location name, Stanford NER and FudanNLP obtain a 0.201 and 0.365 F-measure. Therefore, the research objective in this article is to build a framework for web NER model construction on the web. Note that we had translated testing data from traditional Chinese into simplified Chinese since Stanford and Fudan's build-in models were trained with simplified Chinese training data.

In practice, sometimes there are existing structured databases of known entities that can be utilized for automatic labeling. For example, personal names, organization names, and location names can be obtained from a Who's Who website, yellow-page website, and open government data for registered schools and businesses, respectively. Using these seed names as queries, we can obtain search results from search engines to prepare training examples for many information extraction tasks. Via automatically labeling known entities in the search results, we can obtain a large labeled training set. Although such training data may contain errors, we can further filter unreliable labeling via self-testing.

To further improve the NER performance, we also explore unlabeled training data for semisupervised learning. In this article, we explore the possibility of extending semisupervised learning to sequence labeling via tri-training [Zhou and Li 2005] so that unlabeled training examples can also be used in the learning phase. The challenge

here is to obtain a common label sequence as a consensus answer from two models. As enumerating all possible label sequences will be too time-consuming, we employ a confidence level to control the colabeling answer such that a label sequence with the largest probability is selected. Comparing with a common label sequence from multiple models, the most probable label sequence has a larger chance to obtain a consensus answer for training and testing. Another key issue with tri-training is the assumption of the initial error rate (0.5), leading to a limited number of colabeling examples for training and early termination for large dataset training.

To validate the proposed method, we conduct experiments on three Chinese named entity recognition tasks including personal name, location name, and business organization name. Meanwhile, we also test this framework on Japanese business organization name recognition. Given the seed names for these target entities, we collect search results containing these names from Google's search engine using the known entities as query keywords and automatically label these articles containing known entities. For person name recognition, the proposed approach reaches a 0.911 F-measure on a test set of 8,672 news articles containing 54,546 personal names (11,856 distinct names). For Chinese business organization name recognition, we collect search results with 2,000 known entity queries as our test set. The result shows a 0.849 F-measure on 38,692 sentences (16,241 distinct business organization names). The last three tasks are conducted on smaller test sets using 200 queries to collect test sentences. Finally, the trained models have a performance of a 0.845 F-measure for Chinese location name recognition, a 0.832 F-measure for English business organization name recognition, and a 0.803 F-measure for Japanese business organization name recognition.

The major contributions of our work are fourfold. First, we provide a framework to train an NER model for specific kinds of named entities on the web. Second, we derive a better initialization method for tri-training so that it can be applied to large datasets. Third, we solve the error accumulation problem due to very weak classifiers by including a confidence parameter. Fourth, we conduct five NER tasks and compare with the Stanford NER tool as shown in Table I for Chinese NER tasks. The rest of this article is organized as follows. Section 2 reviews the previous work on NER and semisupervised learning. Section 3 presents our framework. Section 4 reports the experimental results. Finally, Section 5 provides a conclusion and future work.

2. RELATED WORK

Entity extraction is the task of recognizing named entities from unstructured text documents, which is one of the information extraction tasks to test how well a machine can understand the messages written in natural language and automate mundane tasks normally performed by humans. Entity extraction is also important to the semantic web, which promotes common data formats and exchange protocols to allow data to be shared and reused across applications.

Early schemes for named entities relied on rule-based extraction methods with manually coded rules. However, manual coding of rules is tedious and costly. Therefore, algorithms for automatically learning from examples were developed either via rule-based or statistical methods. The former is driven by logical constraints based on hard predicates, whereas the latter makes decisions based on a soft logic from a weighted sum of predicate firing.

In fact, the need to deal with noise in unstructured data has urged the development of machine-learning research from classification to sequence labeling such as Hidden Markov Model (HMM) [Satish and Gururaj 1993], Maximum-Entropy Markov model (MEMM) [McCallum et al. 2000], and Conditional Random Field (CRF) [McCallum and Li 2003]. A more detailed survey on various information extraction methods can be found in Sarawagi's Information Extraction [Sarawagi 2008].

Several tools exist for named entity extraction, including Apache OpenNLP, the Stanford NLP package, and the submodule of General Architecture for Text Engineering (GATE): A Nearly New Information Extraction System (ANNIE). OpenNLP supports maximum entropy (logistic regression) and Pperceptron-based (multilayer Perceptron) machine learning. The Stanford named entity recognizer provides linear chain conditional random field sequence models, particularly for PERSON, ORGANIZATION, and LOCATION. In addition, GATE is a Java suite of tools that can handle up to 12 languages, including English, Chinese, and others. For Chinese, FudanNLP [Qiu et al. 2013] is developed for simplified Chinese NLP and NER.

2.1. Supervised Sequence Labeling Models

Most named entity recognition research has been approached via sequence labeling models based on supervised learning. There are three common sequence models applied to such applications.

The first model is HMM. HMM is a statistical Markov model that is assumed to be a Markov process with unknown (hidden) states. In a Markov model, the state is directly visible to the observer, and its state transition probabilities are the only parameters. In a hidden Markov model, the state is not directly visible to be observed, but its output is visible. In HMM, each state has probability distributions over its possible outputs. HMM can be considered a generalization of a mixture model where the hidden variables are related through a Markov process rather than independent of each other.

The second model is MEMM, or conditional Markov model (CMM) [Borthwick 1999]. MEMM combines features of the hidden Markov model and maximum entropy models. MEMM extends a standard maximum entropy classifier based on assuming the unknown variables are connected in a Markov chain rather than being conditionally independent of each other.

The last model is CRF. CRF is a statistical modeling method based on a discriminative undirected probabilistic graphical model, which can take context into consideration. CRF uses a feature matrix to encode known relationships between observations and to infer unknown variables. It is often used for labeling or parsing of sequential data, such as natural language processing (NLP) or biological sequences, and in computer vision. For example, CRF is often applied in NLP to predict sequences of labels for sequence input samples.

2.2. Distant Learning

Distant supervision learning is a semisupervised learning algorithm that uses weakly labeled training data via a heuristic labeling rule or known small knowledge base. For example, Snow et al. [2005] exploited hand-created regular expressions to automatically extract novel pairs for identifying hypernym (is-a) relations between entities. They used "dependency path" features extracted from parse trees to determine whether two nouns in a news article participate in a hypernym relationship. The classifier with the best performance used additional training data collected from Wikipedia, with an expanded feature lexicon of 200,000 dependency paths to reach a 0.359 F-measure.

As another example, Mintz et al. [2009] provided an alternative paradigm that does not require labeled corpora and allows the use of corpora of any size. They used Freebase [Bollacker et al. 2008], which is a large semantic database and contains several thousand known relations to provide distant supervision. For each relation pair of entities that occur one or more times in Freebase, they found all sentences containing those entities in a large unlabeled corpus and extracted related features to train a relation classifier. They combined the advantages of supervised IE (combining 400,000 noisy pattern features in a probabilistic classifier) and unsupervised IE (extracting large

numbers of relations from large corpora of any domain) to extract 10,000 instances of 102 relations at a precision of 0.676.

As the starting knowledge base, also called a reference set, is important for information extraction from unstructured text, Michelson and Knoblock [2009] proposed an approach to building the reference set directly from the text itself. Starting with a small amount of background knowledge, the technique constructs tuples representing the entities in the text to form a reference set. They evaluated their system on nine information extraction tasks including attribute names for cars, laptops, and skis from posts on Craigslist. The results showed their method achieved an improvement in F-measure for six recognition entity categories (total nine categories) and was competitive in performance on the others.

An et al. [2003] used known seed entities as query keywords to collect web pages that contain the entity instances. This automatically tagged corpus may have lower quality than the manually tagged ones, but its size can be almost infinitely increased without any human effort. An et al. used about 28 to 51 times of automatically tagged named entities and got almost a similar performance (0.85) as manually tagged ones for Korean person names. Rae et al. [2012] proposed a similar method to increase the amount of training data by bootstrapping from web snippets. They used Wikipedia titles and social media check-in tag names (Foursquare and Gowalla) as known POIs to query search engines to collect training sentences. With 10-fold cross-validation, they could identify POIs in free text with an F-measure between 0.557 and 0.915. Fu et al. [2011] proposed an approach to generate large-scale Chinese NER training data from an English-Chinese discourse-level aligned parallel corpus. They trained a high-performance NER system to label named entities in the English corpus and labeled the Chinese corpus according to the word-level alignment and bilingual corpus. In their experiments, they generated a Chinese NER corpus with 167,100 sentences with a 67.89% F-measure for the 863-Evaluation corpus and a 73.20% F-measure for the OntoNotes corpus.

2.3. Semisupervised Learning

The term semisupervised (or weakly supervised) learning involves a small degree of supervision to avoid the high cost of obtaining large labeled training examples. Semisupervised systems discover similar instances of known examples and retrain the model to improve the performance. By repeating this process, a large number of labeled examples will eventually be gathered for model training. Therefore, semisupervised learning also refers to techniques that also make use of unlabeled data for training. Many approaches have been previously proposed for semisupervised learning, including self-learning (bootstrapping) [Riloff et al. 2003], an Expectation-Maximization-based approach, a graph-based approach, a semisupervised support vector machine (S^3VM) [Bennett and Demiriz 1999], cotraining [Blum and Mitchell 1998], and tri-training [Zhou and Li 2005].

In self-training approaches, a classifier is first trained on the labeled instances and then applied to unlabeled instances. Some subset of those newly labeled instances is then used (in conjunction with the original labeled instances) to retrain the model. EM-based approaches train an initial model using just labeled examples via Maximum Likelihood Estimation (MLE) or Maximum A Posterior (MAP) Estimation, then use this model to "guess" the labels of unlabeled examples for model retraining. The process repeats until it converges. Graph-based approaches represent each example (labeled/unlabeled) as vertices of some graph and apply an idea called graph-based regularization for prediction function optimization. Graph-based approaches exploit the property of label smoothness in the optimization design: they not only minimize the loss on labeled data but also ensure the label smoothness of both labeled and

unlabeled data along the graph. Semisupervised SVM (S^3VM), or transductive SVM (TSVM), adds a constraint to the original SVM optimization function in order to preserve the margin over unknown test labels. Approximation search must be undertaken to avoid brute-force search over all possible labelings.

A broad survey of semisupervised learning methods [Chapelle et al. 2006] found that semisupervised approaches do not uniformly beat supervised methods and that there is no clear winner from among the methods. However, the semisupervised learning performance is very close to a supervised system with full data as reported in Yu and Kubler [2010], who applied an EM-based naive base classifier (EM-NB), S^3VM, self-training, and cotraining for opinion detection tasks.

Although a number of semisupervised classifications have been proposed, semisupervised learning for sequence labeling has received considerably less attention and is designed according to a different philosophy. For example, Ando and Zhang [2005] employed a learning paradigm to create auxiliary problems automatically from unlabeled data so that predictive structures can be learned from the common structure shared by the multiple classification problems, which can then be used to improve performance on the target problem. They used the testing data of CoNLL'03 named entity chunking (English and German) to evaluate their method, which achieved a 0.893 (English) and 0.753 (German) F-measure. In addition, they also applied their method on a CoNLL'00 syntactic chunking task, and an F-measure of 0.944 (all) and 0.947 (noun phrases only) could be achieved.

In addition to using both labeled and unlabeled data, Mann and McCallum [2010] presented a semisupervised training method using weakly labeled data for maximum entropy models and conditional random fields. This method penalizes models by divergence between the model's expectations over the unlabeled data and input conditional probabilities. Their work showed that CRF trained with label regularization outperforms alternative methods and achieves a 1% to 8% performance improvement over supervised-only approaches.

2.4. Cotraining-Based Learning

Cotraining and tri-training have been mainly discussed for classification tasks with relatively few labeled training examples. For example, the original cotraining paper by Blum and Mitchell [1998] described experiments to classify web pages into two classes using only 12 labeled web pages as examples via cotraining. This cotraining algorithm requires two views of the training data and learns a separate classifier for each view using labeled examples. Nigam and Ghani [2000] demonstrated that cotraining performed better when the independent feature set assumption is valid. For comparison, they conducted their experiments on the same (WebKB course) dataset used by Blum and Mitchell.

Goldman and Zhou [2000] relaxed the redundant and independent assumption and presented an algorithm that uses two different supervised learning algorithms to learn a separate classifier from the provided labeled data. Empirical results demonstrated that two standard classifiers can be used to successfully label data for each other with a 95% confidence interval.

Tri-training was an improvement of cotraining, which used three classifiers and a voting mechanism to solve the confidence issue of colabeled answers by two classifiers. In each round of tri-training, the classifiers h_j and h_k choose some examples in (U) to label for h_i $(i, j, k \in \{1, 2, 3\}, i \neq j \neq k)$. Let L_i^t denote the set of examples that are labeled for h_i in the tth round. Then the training set for h_i in the tth round is $L \cup L_i^t$. Note that the unlabeled examples labeled in the tth round, that is, L_i^t, won't be put into the original labeled example set, that is, (L). Instead, in the $(t + 1)$-th round, all

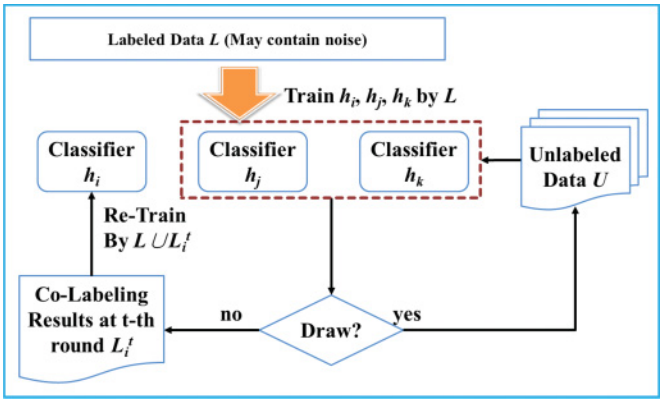


Fig. 1. Tri-training.

the examples in L_i^t will be regarded as unlabeled and put into (U) again, as shown in Figure 1.

There are several ways to construct the initial three classifiers. While bootstrapped sampling was used in Zhou and Li [2005], researchers have tried various learning algorithms and multiple views as well as their combinations for initial classifier construction to achieve the best precision of 0.804 [Nguyen et al. 2008].

3. SYSTEM ARCHITECTURE

Due to the high cost of labeling, most benchmarks for NER are limited to several thousand sentences. Two intuitive ways are considered in this article: one is automatic labeling of unlabeled data for preparing a large amount of annotated training examples, and the other is semisupervised learning for making use of both labeled and unlabeled data during learning. For the former, automatic labeling is sometimes possible, especially for named entities that can be obtained from web resources. For example, person names can be found from Who's Who websites; location names can be obtained from government open data or travel websites; business organization names can be collected from yellow-page websites; and so forth. For the latter, tri-training is extended to sequence labeling with a proper design of selecting the label sequence for unlabeled examples during colabeling. Furthermore, we also modify the initialization step of tri-training to deal with large training sets.

As shown in Figure 2, we propose a framework for web NER model construction. Given a list of entity names as input, the system first fetches entity name results from Google search engines to prepare labeled training examples. This includes steps of automatic labeling, feature engineering, and self-testing. The second part is the semisupervised learning based on tri-training.

3.1. Training Data Preparation

Given known entity names, we can prepare training examples by querying search engines and conducting automatic labeling on search results. Using a known entity as a search keyword, we will obtain search snippets containing these entity names. Depending on the web corpus that the NER model is designed for, we can specify the search scope, that is, the websites where the search results are obtained.

We use three different automatic labeling methods to label known named entities in the collected sentences. The first method is uni-labeling, which uses the query input itself to label occurrences of the entity in the collected sentences, but this method does

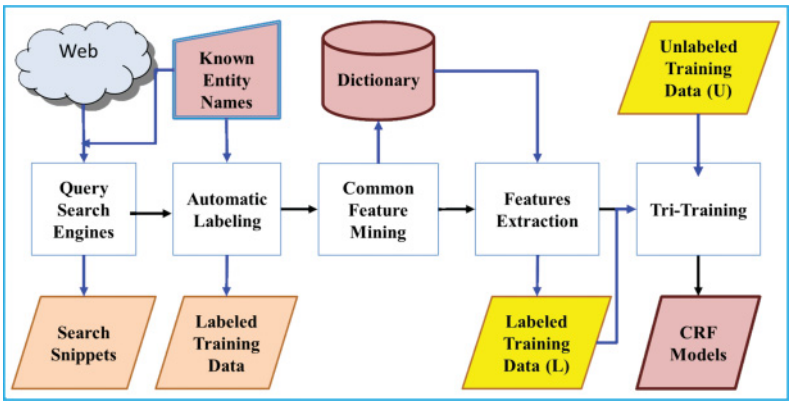


Fig. 2. System architecture.

not label any other known entities. The second method is self-labeling, which uses all query entities to label collected data. The third method is full-labeling, which uses not only the query inputs but also the entire known entities for annotation. In our experiment, we use different numbers (DS1 = 500, DS2 = 1,000, $\cdots$ DS6 = 7,053) of known entities to demonstrate the learning curve. For DS1, we have 500 query entities for self-labeling, but the entire entities number 7,053.

For each occurrence of an entity, we keep 20 words before and after the entity to form a sentence since shorter sentences can speed up the training process, especially when there are a lot of training examples. To increase the consistency of the boundary features, the preprocessing procedure also transforms all kinds of 2-byte brackets into 1-byte left parenthesis and replaces addresses, telephone numbers, and time and dates with special symbols via regular expression matching.

For feature extraction, each feature will represent whether a given token possesses a certain attribute. For example, if a token is a digit, a punctuation, or a POS tag, then the feature value sets 1, or 0 if not. Thus, a general approach is to use a dictionary to list a set of keywords presenting such an attribute, which requires language expertise to locate such keywords. For example, we may use internal features that frequently occur in a named entity as well as contextual features surrounding a named entity to recognize named entities.

To enable the construction of web NER models for different languages, we conduct frequent prefix (and postfix) mining on named entities to enumerate frequent 1-, 2-, and 3-prefix (postfix) as dictionaries for internal feature extraction. Similarly, we also mine frequent postfix (prefix) on strings before (after) named entities to enumerate frequent 1-, 2-, and 3-patterns as dictionaries for contextual feature extraction. In addition to these designed features, part-of-speech (POS) tags and punctuation symbols are also the default features. Table II lists the default features used for our tool and the corresponding examples for English organization entity recognition.

3.2. Self-Testing

Since automatic annotation may involve wrong labeling, we also apply self-testing to filter low-confidence examples. That is to say, we assume sentences with lower confidence to be noisy and filter them to maintain a dataset with better quality. Here, we suppose the output probability of the sequence labeling model is the confidence score.

We first train a model with the automatically labeled training data and then use the trained model to test the training data as shown in Figure 3. Assuming sentences with

Table II. Automatic Constructed Dictionary for English Business Organization Recognition

Feature	Examples
Length 1 Entity prefix	American, national, united, university
Length 2 Entity prefix	association of, house of, institute of
Length 3 Entity prefix	national association of, university of the
Length 1 Entity postfix	company, corporation, group, incorporation
Length 2 Entity postfix	computer incorporation, technology group
Length 3 Entity postfix	management co., manufacturing company limited
Length 1 Common words before entity	at, for, in, the
Length 2 Common words before entity	go to, work at, work for
Length 3 Common words before entity	allow to visit, work together at
Length 1 Common words after entity	for, in, is, with
Length 2 Common words after entity	focus on, lead to, with nearby
Length 3 Common words after entity	in advance to, lead to develop
Part-of-speech (POS) tags	N, V, NP
Symbol	":", "(", ")"

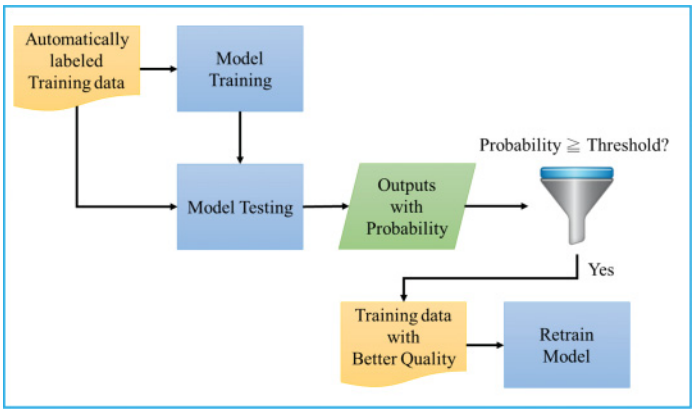


Fig. 3. Self-testing flow chart.

lower probability are noise, we remove sentences with confidence less than a threshold α and use the remaining ones to retrain a model with less data and better data quality. In a way, the noise that occurs during automatic labeling could be complemented by the self-testing procedure. This self-testing process therefore improves the quality of training data. Overall, the benefit of the large amount of labeled training examples is greater than the noise it may cause.

3.3. Tri-Training

Let L denote the labeled example set with size $|L|$ and U denote the unlabeled example set with size $|U|$. In each round, t, tri-training uses two models, h_j and h_k, to label the answer of each instance x from unlabeled training data U. If h_j and h_k give the same answer, then we could use x and the common answer pair as a new training example, that is, $L_i^t = \{(x, y) : x \in U, y = h_j^t(x) = h_k^t(x)\}$ for model h_i $(i, j, k \in \{1, 2, 3\}, i \neq j \neq k)$. To ensure that the error rate is reduced through iterations, when training h_i, Equation (1) must be satisfied:

$$e_i^t \left|L_i^t\right| < e_i^{t-1} \left|L_i^{t-1}\right|, \tag{1}$$

where e_i^t denotes the error rate of model h_i in L_i^t, which is estimated by h_j and h_k in the tth round using the labeled data L by dividing the number of labeled examples on

which both h_j and h_k make an incorrect estimation by the number of labeled examples for which the estimation made by h_j is the same as that made by h_k, as shown in Equation (2):[1]

$$e_i^t = \frac{\left|\{x, y\} \in L, h_j^t(x) = h_k^t(x) \neq y\right|}{\left|\{x, y\} \in L, h_j^t(x) = h_k^t(x)\right|}. \tag{2}$$

If $|L_i^t|$ is too large, such that Equation (1) is violated, it would be necessary to sample maximum u examples from L_i^t such that Equation (1) can be satisfied. Let integer u denote the upper bound of new training examples in the tth round and S_i^t denote the subset of L_i^t. We add a small value ϵ in the numerator and denominator of Equation (3):

$$u = \left\lceil \frac{\left(e_i^{t-1} + \epsilon\right)\left|S_i^{t-1}\right|}{e_i^t + \epsilon} - 1 \right\rceil \tag{3}$$

$$S_i^t = \begin{cases} Subsample\left(L_i^t, u\right) & \text{violate Equation (1)} \\ L_i^t & \text{otherwise.} \end{cases} \tag{4}$$

For the last step in each round, the union of the labeled training examples L and S_i^t, that is, $L \cup S_i^t$, is used as training data to update classifier h_i for this iteration.

3.3.1. Modification for the Initialization. According to Equation (1), the product of error rate and the size of new training examples define an upper bound for the next iteration. Meanwhile, $|L_i^{t-1}|$ should satisfy Equation (5) such that $|L_i^t|$ after subsampling, that is, u, is still bigger than $|L_i^{t-1}|$:

$$\left|L_i^{t-1}\right| > \frac{e_i^t}{e_i^{t-1} - e_i^t}. \tag{5}$$

In order to estimate the size of $|L_i^1|$, that is, the number of new training examples for the first round, we need to estimate e_i^0, e_i^1, and $|L_i^0|$ first. Zhou et al. assumed a 0.5 error rate for e_i^0, computed e_i^1 by h_j and h_k, and estimated the lower bound for $|L_i^0|$ by Equation (5); thus:

$$\left|L_i^0\right| = \left\lfloor \frac{e_i^1}{e_i^0 - e_i^1} + 1 \right\rfloor = \left\lfloor \frac{e_i^1}{0.5 - e_i^1} + 1 \right\rfloor. \tag{6}$$

The problem with this initialization is that, for a larger dataset L, such an initialization will have no effect on retraining and will lead to an early stop for tri-training. For example, consider the case when the error rate e_i^1 is less than 0.4; then the value of $|L_i^0|$ will be no more than 5, leading to a small upper bound for $e_i^1|L_i^1|$ according to Equation (1). That is to say, we can only sample a small subset S_i^1 from L_i^1 for training h_i based on Equation (4). On the other hand, if e_i^1 is close to 0.5 such that the value of $|L_i^0|$ is greater than the original dataset $|L|$, it may completely alter the behavior of h_i.

To avoid this difficulty, we propose a new estimation for the product $e_i^0|L_i^0|$. Let $L^C(h_j, h_k)$ denote the set of labeled examples (from L) on which the classification made by h_j is the same as that made by h_k in the initial round, and $L^W(h_j, h_k)$ denote the set of examples from $L^C(h_j, h_k)$ on which both h_j and h_k make an incorrect classification,

[1]Assuming that the unlabeled examples hold the same distribution as that held by the labeled ones.

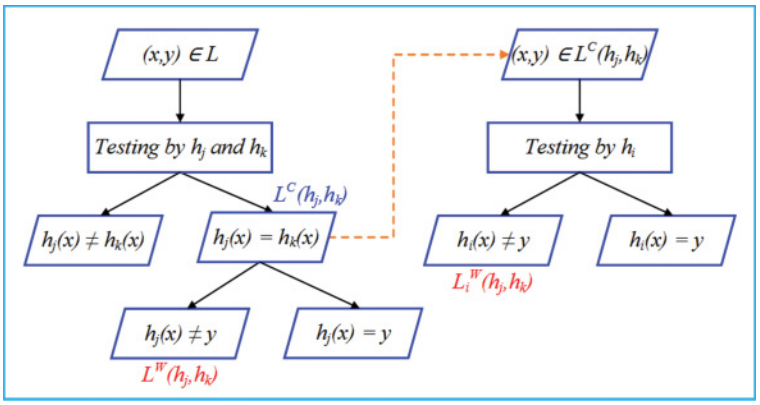


Fig. 4. The relationship among Equations (7), (8), and (9).

as shown in Equations (7) and (8). In addition, we define $L_i^W(h_j, h_k)$ to be the set of examples from $L^C(h_j, h_k)$ on which h_i makes an incorrect classification in the initial round, as shown in Equation (9). The relationship among $L^C(h_j, h_k)$, $L^W(h_j, h_k)$, and $L_i^W(h_j, h_k)$ is illustrated in Figure 4.

$$L^C(h_j, h_k) = \{(x, y) \in L : h_j(x) = h_k(x)\} \tag{7}$$

$$L^W(h_j, h_k) = \left\{(x, y) \in L^C(h_j, h_k) : h_j(x) \neq y\right\} \tag{8}$$

$$L_i^W(h_j, h_k) = \left\{(x, y) \in L^C(h_j, h_k) : h_i(x) \neq y\right\} \tag{9}$$

By replacing $e_i^0|L_i^0|$ with $L_i^W(h_j, h_k)$ and estimation of e_i^1 by $|L^W(h_j, h_k)|/|L^C(h_j, h_k)|$, we can estimate an upper bound for $|L_i^0|$ via Equation (3). That is to say, we can compute an upper bound for $|L_i^0|$ and replace Equation (3) with Equation (10) to estimate the maximum data size of $|L_i^1|$, in the first round. Similarly, we add one to $L_i^W(h_j, h_k)$ and $L^W(h_j, h_k)$ respectively to avoid division by zero.

$$\left|L_i^1\right| = \left\lceil \frac{e_0^0 \left|L_i^0\right|}{e_i^1} - 1 \right\rceil = \left\lceil \frac{\left(\left|L_i^W(h_j, h_k)\right| + 1\right) * \left|L^C(h_j, h_k)\right|}{\left|L^W(h_j, h_k)\right| + 1} - 1 \right\rceil \tag{10}$$

3.3.2. Limited Upper Bound. It is worth noting that the reduced error for the second round could admit a lot of new examples in a larger dataset according to Equation (3). In some cases, the amount could be much larger the original data set, dominating the performance of the re-trained classifier. As shown in Table VII in the appendix, the value u in round 2 is almost 3 times the size of $|L|$. To avoid such dominance, we propose $|L| + |S_i^{t-1}|$, the size of previous training data as a bound for this round. Therefore, we take the minimum of $|L| + |S_i^{t-1}|$ and u in Equation (3) as our limited upper bound, see Equation (11) below.

$$u = \min\left(|L| + \left|S_i^{t-1}\right|, u\right). \tag{11}$$

3.3.3. Modification for Colabeling. The tri-training algorithm was originally designed for traditional classification. For sequence labeling, we need to define what should be the common labels for the input example x when two models (training time) or three models (testing time) are involved.

While tri-training has been used in many classification tasks, the application in sequence labeling tasks is limited. Chen [2006] considered only the most probable label

sequence from each model and used the agreement measure as shown in Equation (12) to select examples for h_i with the highest-agreement sentences labeled by h_j and h_k, and the lowest-agreement sentences labeled by h_i and h_j. This method was named Two Agree One Disagree (S2A1D):

$$A_g(y_j, y_k) = \frac{\sum_{1 \le d \le n} I(y_{jd} = y_{kd})}{n}, \tag{12}$$

where $y_j = \{y_{j1}, y_{j1}, y_{j1} \dots y_{jn}\}$ is the label result by h_j with length $|y_j| = n$. The new training samples are chosen to be the ones that are labeled by h_j, ignoring the label result by h_k. A control parameter is used to determine the percentage (30%) of examples selected for the next round. The process iterates until no more unlabeled examples are available. Thus, Chen et al.'s method does not ensure the probably approximately correct learning (PAC learning) theory [Valiant 1984].

As the probability for two sequence labelers to output the same label sequence is low (a total of $5^{|l|}$ possible label sequences with BIESO tagging and length l), in this article, we propose a different method to resolve this issue. Assume that each model can output the m best label sequences with the highest probability ($m = 3$). Let $P_i(y \mid x)$ denote the probability that an instance x has label y estimated by h_i. We select the label with the largest probability sum by the colabeling models. In other words, we could use h_j and h_k to estimate possible labels, then choose the label y with the maximum probability sum, $P_j(y \mid x) + P_k(y \mid x)$, to retrain h_i. Thus, the set of examples, L_i^t, prepared for h_i in the tth round is defined as follows:

$$L_i^t = \left\{ (x, y) : x \in U, \max_y (P_j(y|x) + P_k(y|x)) \ge \theta * 2 \right\}, \tag{13}$$

where θ (default 0.5) is a threshold that controls the quality of the training examples provided to h_i.

During testing, the label y for an instance x is determined by three models h_1, h_2, and h_3. We choose the output with the largest probability sum from n models with a confidence $\theta * n$ for $n = 3$ and 2. If the label with the largest probability sum from n models is not greater than $\theta * n$, then we choose the one with the largest probability from a single model with a maximum probability. That is to say, if the label with the largest probability sum from three models is not greater than $\theta * 3$, then we choose the one with the largest probability sum from two models with a confidence of $\theta * 2$. The last selection criterion is the label with the maximum probability estimated by the three models as shown in Equation (14). Finally, the complete tri-training algorithm for sequence labeling is shown in Algorithm 1 in the appendix.

$$y = \max_y \begin{cases} \max_y (P_1(y|x) + P_2(y|x) + P_3(y|x)) \ge \theta * 3 \\ \max_y (P_i(y|x) + P_j(y|x)) \ge \theta * 2, i, j \in \{1, 2, 3\}, i \ne j \\ \max_y (P_1(y|x), P_2(y|x), P_3(y|x)) \end{cases} \tag{14}$$

4. EXPERIMENTS

We first apply our framework on the Chinese personal name extraction task and evaluate the recognition performance with different parameter settings. Second, we apply our proposed approach on four other NER tasks in Chinese, English, and Japanese in Cection 4.2. We use the CRF++ [Kudo 2005] package for model training. We define the Precision, Recall, and F-measure based on the number of entity names as follows:

$$Precision = \frac{|Correctly\ identified\ entities|}{|Identified\ entities|} \tag{15}$$

Table III. Chinese Person Name Training Data

	L						U
	DS1	DS2	DS3	DS4	DS5	DS6	–
#Seed Names	500	1,000	2,000	3,000	5,000	7,053	–
#Sentences	5,548	10,928	21,267	30,653	50,738	67,104	240,995
#Tokens	106,535	208,383	400,111	567,794	913,516	1,188,822	4,251,861

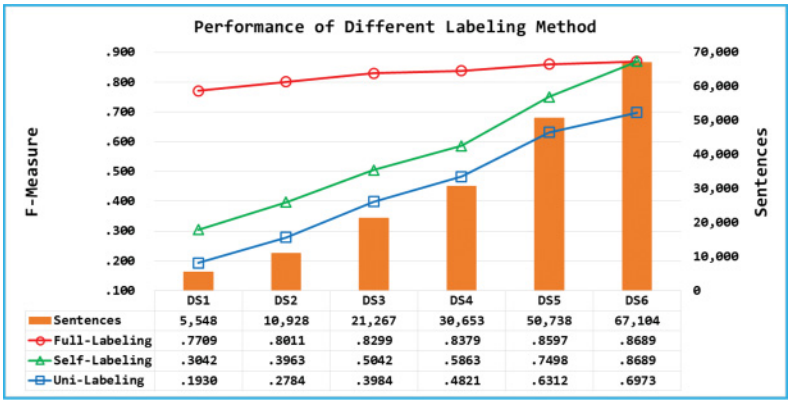


Fig. 5. Performance of different labeling methods.

$$Recall = \frac{|Correctly\ identified\ entities|}{|Real\ entities|} \tag{16}$$

$$F - measure = \frac{2 * Precision * Recall}{Precision + Recall}. \tag{17}$$

4.1. Chinese Person Name Extraction

We use known celebrity names to query search engines and collect the top 10 URLs in four news domains, including Liberty Times, Apple Daily, China Times, and United Daily News, for sentences that contain the query keywords. Note that the collected pages contain a lot of reporter names and could not be labeled by full-labeling. Therefore, we also include user-defined rules to label reporter names to maintain data quality. The dictionaries' contents are reviewed by humans and contain 486 job titles, 224 surnames, 38,261 first names, and 107 symbols, as well as 223 common characters and words. To show the effectiveness of training data size, we split the training dataset with a different number of seed entities as shown in Table III.

For unlabeled and testing data, we crawl the same news websites from January 1, 2013, to March 31, 2013, to obtain 20,974 articles. To increase the possibility of containing person names, we select sentences that include some common surnames followed by some common first names to obtain 240,994 sentences as unlabeled data U. For testing, we manually labeled 8,672 news articles from 11 categories including politics, finance, sports, and entertainment, and yielding a total of 364,685 sentences with 54,546 person names (11,856 distinct person names).

4.1.1. Automatic Labeling. We first show the performance of three different labeling methods with a different number of training sizes. As shown in Figure 5, uni-labeling has very limited performance, while self-labeling presents real performance when a fixed number of known entities is available. For example, if we have only 500 celebrity names (DS1), we can only achieve a 0.304 F-measure with 5,548 sentences via

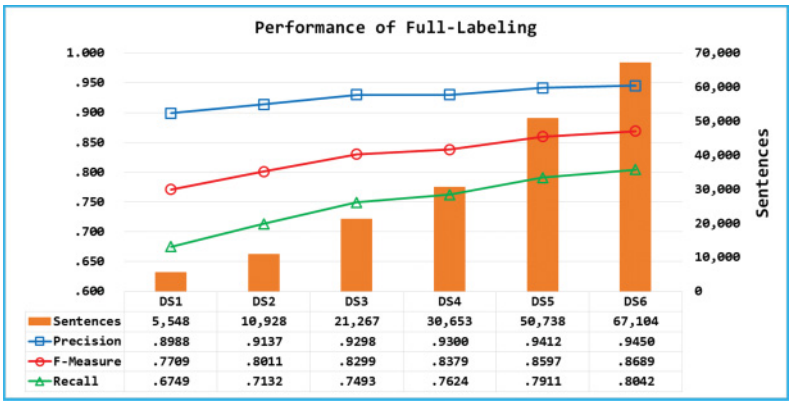


Fig. 6. Performance of full-labeling.

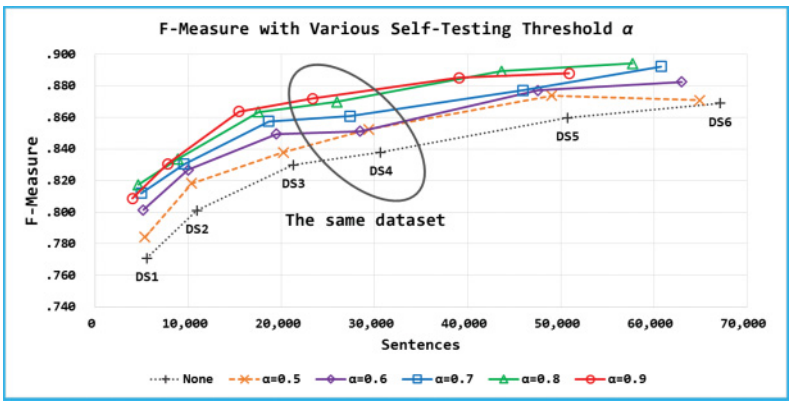


Fig. 7. F-measure with various self-testing thresholds.

self-labeling. However, via full-labeling, DS1 is enough to build a model with a 0.771 F-measure. If we have 7,000 known entities (DS6), the model can achieve a 0.869 F-measure via full-labeling, which echoes the claim that more training data can improve the model. As we can see, the precision reaches a high value of 0.945, while the recall is only 0.804; thus, there is still some room for improvement (Figure 6).

4.1.2. Self-Testing Performance. Since the automatic labeling procedure does not ensure perfect training data, we design a self-testing procedure to filter examples with low confidence and retrain a new model with the set of high-confidence examples. The idea is to use the trained CRF model to test the training data themselves and output the conditional probability for the most possible label sequence. By setting various thresholds, we obtain a dataset with a different number of training examples. If the threshold is too high, we may filter too many sentences and are left with fewer training examples. Figure 7 shows the change of F-measure and the size of training examples with respect to the change of the confidence threshold for the six datasets.

The performance of self-testing is improved for all datasets with confidence levels from 0.5 to 0.9. The best performance is achieved at a confidence level of 0.8 for all datasets except DS3 and DS4, which have the best performance when the threshold is 0.9. Overall, the effect of increasing data quality is larger than the effect of reducing data size.

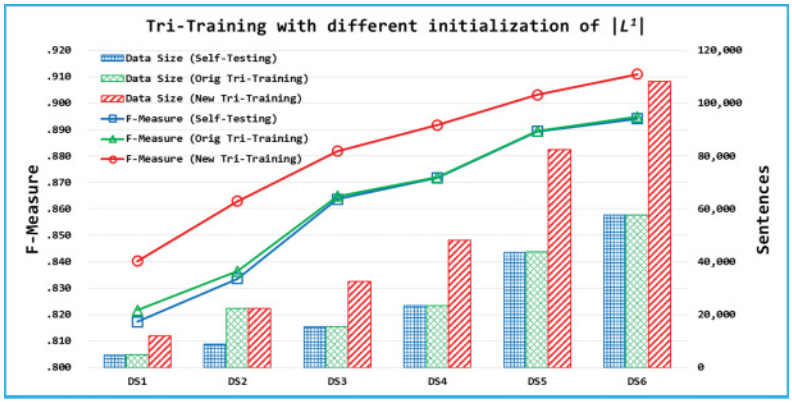


Fig. 8. Tri-training with different initializations $|L_i^1|$.

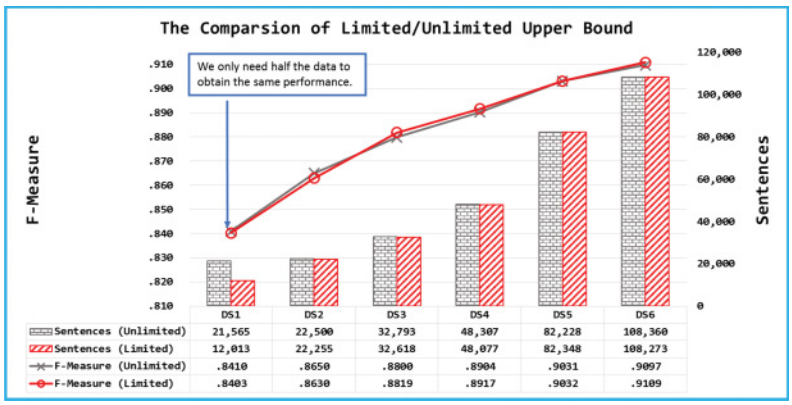


Fig. 9. The comparison of limited and unlimited upper bounds.

4.1.3. Tri-Training Performance. In our initial attempt to apply original tri-training, we obtained no improvement for all datasets. As shown in Figure 8, the performance and the final data size used for training are similar to those values obtained for the self-testing results reported earlier. This is because we have a very small estimation $|L_i^0|$ by Equation (6) when a 0.5 initial error rate for e_i^0 is assumed. Therefore, it does not have any improvement on retraining.

However, with the new initialization by Equation (10), the number of examples $|L_i^1|$ is greatly increased. For DS1, the unlabeled data selected is five times the original data size (an increase from 4,637 to 12,013), leading to an improvement of 0.023 in F-measure (from 0.817 to 0.840). For DS2, the final data size is twice the original data size (from 8,881 to 22,255) with an F-measure improvement of 0.029 (from 0.834 to 0.863). For DS6, the improvement in F-measure is 0.017. Overall, an improvement of 0.014 to 0.029 can be obtained with this tri-training algorithm.

To avoid the domination of unlabeled data, we include Equation (11) to set an upper bound for new training data in each round. The results in Figure 9 show that we only need half the data to obtain the same performance for the smaller dataset (DS1), while the upper bound has no effect for larger datasets. Figure 10 shows the number of iterations executed for 300 tri-training experiments with a colabeling threshold θ between 0.5 and 0.9 (half with and half without upper-bound setting), where most

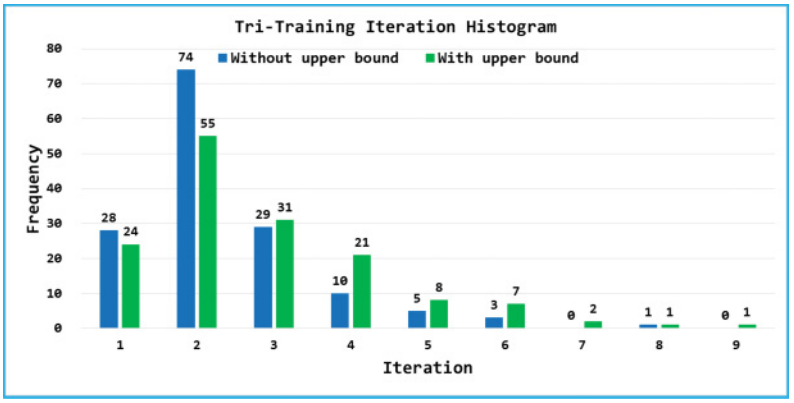


Fig. 10. Tri-training iteration histogram.

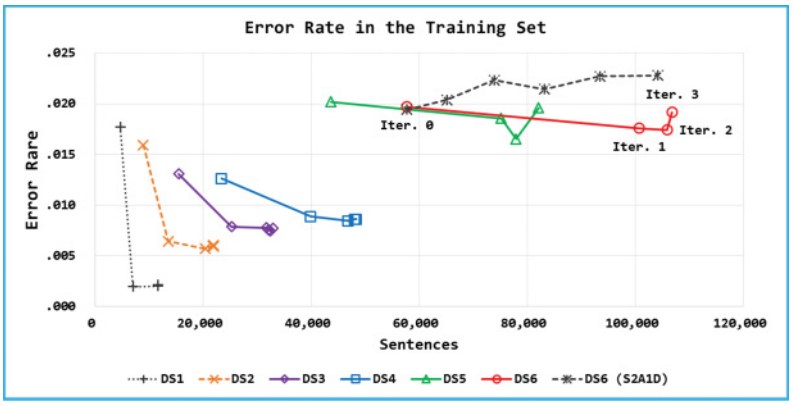


Fig. 11. Error rate in the training set.

tri-training experiments run less than or equal to three rounds. The average number of iterations is increased from 2.36 to 2.83, and the number of examples used for the final iteration is decreased from 52,397 to 50,233. Therefore, the upper-bound setting effectively reduces the number of training examples used for tri-training while maintaining a similar performance.

Next, we compare Chen's selection method, S2A1D, with our proposed colabeling method using the error rate in the training set and the F-measure in the testing set. As shown in Figure 11, the proposed method based on the most probable sequence (MPS) stops (after two or three iterations) when the error rate increases for all datasets, while S2A1D continues execution as long as there are unlabeled training examples. Thus, the error rate fluctuates (i.e., it may increase or decrease) around 0.02. In other words, the modified tri-training by Chen et al. does not follow the PAC learning theory, and there is no guarantee that the performance will increase or decrease in the testing set.

As shown in Figure 12, the F-measure of the MPS in the testing set presents an upward trend, and the three classifiers may outperform each other during different iterations. As for S2A1D, one of the classifiers (h2) continues to be dominated by the other two classifiers and performs worse than the baseline (self-testing). It seems the final output does not take advantage of the three classifiers and has a similar result as the baseline (self-testing). On the contrary, the proposed method could combine opinions of three classifiers and output credible labels via tri-training.

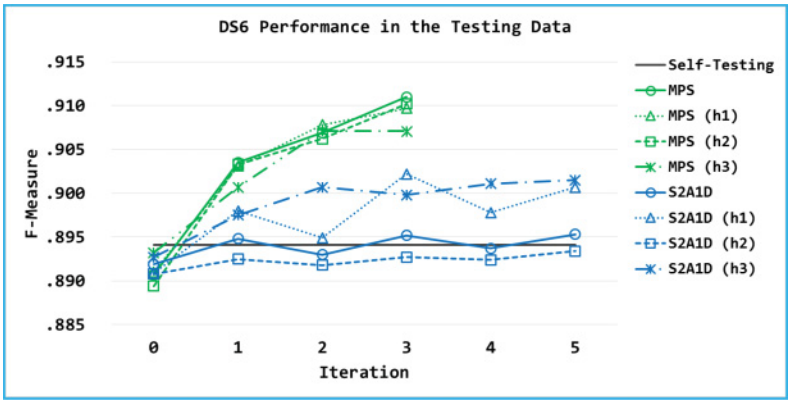
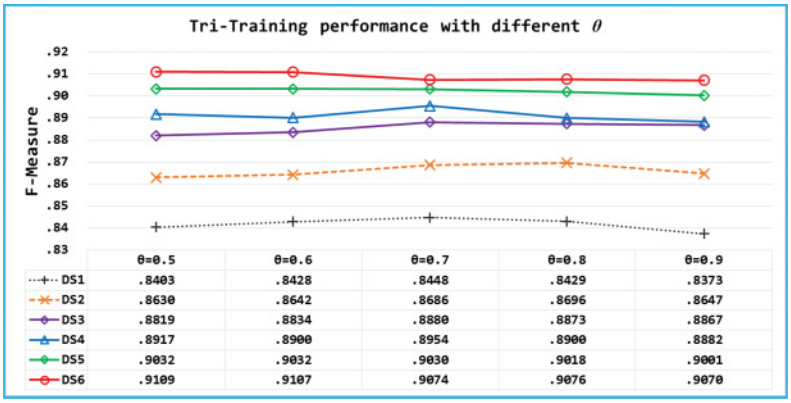


Fig. 12. DS6 performance in the testing data.



	θ=0.5	θ=0.6	θ=0.7	θ=0.8	θ=0.9
DS1	.8403	.8428	.8448	.8429	.8373
DS2	.8630	.8642	.8686	.8696	.8647
DS3	.8819	.8834	.8880	.8873	.8867
DS4	.8917	.8900	.8954	.8900	.8882
DS5	.9032	.9032	.9030	.9018	.9001
DS6	.9109	.9107	.9074	.9076	.9070

Fig. 13. Performance with different θ.

Table IV. Incorrect Person Name Extraction Examples

ID	Example (underline means correct entity)	Extraction
1	體重 140 公斤的戎祥 與老婆現身微風超市採買	None
2	MV 僅輸南韓大叔PSY 騎馬舞	None
3	布萊恩 強調 5 連敗代表球隊的確問題重重	None
4	據說為戰國武將「山梨之虎」武田信玄 家臣的後代	田信玄
5	並要求市長陳菊 先提出財政改革專案報告	陳菊先
6	吳尊 則說自己做過汶萊美食蝦餅、咖哩炒飯秀廚藝	吳尊則

Finally, we change the colabeling threshold θ to observe performance changes. Figure 13 shows that this procedure can help tri-training to achieve better performance. For DS3, the percentage of improvement from self-testing to tri-training in F-measure is 33.90% (0.8637 → 0.8819 vs. 0.8637 → 0.8880). Overall, small datasets (DS1–DS4) need larger θ to ensure data quality to improve tri-training performance to 18.88% to 33.90%. However, this procedure almost cannot improve any performance on larger datasets (DS5 and DS6).

4.1.4. Error Analysis. We present some examples as shown in Table IV to discuss incorrect extraction cases. In the first case, the surname dictionary does not contain the rare surname "戎"; therefore, the NER model misses the extraction of the person name (even though "的"/"與" is a common single word before/after a person name). For the same

Table V. Four NER Tasks' Training and Testing Data

Category	Chinese Location	Chinese Business Org.	English Business Org.	Japanese Business Org.
Data Source	Open Data	Chunghwa Yellow Pages	Yelp	iタウンページ
Training: L (#Query)	10,000	11,138	10,000	10,000
#Sentences	53,313	87,916	39,798	29,999
Training: U (#Query)	30,000	50,000	30,000	30,000
#Sentences	132,486	156,822	100,182	88,074
Testing (#Query)	200	2000	200	200
#Sentences	2,638	38,692	941	809
#Distinct Entities	600	16,241	465	438

reason, transliteration of foreign stars like "布萊恩" (Bryant) and "PSY" are missed because the terms "大 叔"/"騎馬" are not common words before/after a person name. In the fourth sentence, the NER model extracts only a partial name since "田" is a common surname in Chinese, while the term "武田" is a common Japanese surname, which is not included in our dictionary. For the last two sentences, our model extracts an extra token after the correct entity since most Chinese person names in Taiwan are of length 3.

4.2. Other Named Entity Recognition Tasks

Next, we apply our framework on four NER tasks to extract location names in Chinese and business organization names in Chinese, Japanese, and English. We collect location entities from the Taiwan Government Open Data Platform[2] and business organization names from Chunghwa Yellow Pages,[3] iタウンページ.,[4] and Yelp.[5] Next, we use known entities as query keywords and collect the top five search snippets for sentences that contain query keywords. Then, we use the known entities to label matched entities as extraction targets in training data. For unlabeled and testing data, we use the same method to collect sentences from search results with additional queries (see Table V). We remove sentences that are duplicated in our collected data.

For location name and organization name recognition, we redefine the evaluation metrics because some entity names are much longer. For example, 華碩電腦股份有限公司 (ASUSTEK COMPUTER Inc.), 華碩電腦 (ASUSTEK COMPUTER), and 華碩 (ASUSTEK) all refer to the same business entity. If we use exact match, we may miss incomplete occurrences of the longer entity name. For this reason, we redefine the precision and recall scores for each occurrence and average the values for overall precision and recall as follows.

$$P_x = \frac{|Overlap\ tokens|}{|Identified\ entity\ tokens|} \tag{18}$$

$$Precision = \frac{\sum P_x}{|Identified\ entities|} \tag{19}$$

$$R_x = \frac{|Overlap\ tokens|}{|Real\ entity\ tokens|} \tag{20}$$

[2]http://data.gov.tw.
[3]http://www.iyp.com.tw.
[4]http://itp.ne.jp.
[5]http://www.yelp.com.

Table VI. The Performance Comparison of Full-Labeling, Self-Testing, and Tri-Training

Category	Steps	Precision	Recall	F-measure
Chinese Location	Full-Labeling	0.896	0.769	0.828
	Self-Testing	0.900	0.776	0.833
	Tri-Training	0.925	0.777	0.845
Chinese Business Org.	Full-Labeling	0.850	0.779	0.813
	Self-Testing	0.808	0.859	0.833
	Tri-Training	0.825	0.875	0.849
English Business Org.	Full-Labeling	0.781	0.835	0.807
	Self-Testing	0.774	0.868	0.818
	Tri-Training	0.789	0.881	0.832
Japanese Business Org.	Full-Labeling	0.824	0.730	0.774
	Self-Testing	0.841	0.745	0.789
	Tri-Training	0.845	0.766	0.803

$$Recall = \frac{\sum R_x}{|Real\ entities|}. \tag{21}$$

Table VI presents the performance comparison of four NER tasks from the basic model with automatic labeling and self-testing with confidence threshold 0.7, to modified tri-training with the new initialization method and fixed colabeling threshold θ 0.5. Overall, the performance is not as good as Chinese person name recognition as the dictionaries are all constructed automatically via frequent prefix and postfix mining on named entities (as well as their contexts) to prepare the dictionaries for feature extraction. We also notice that most NER tasks have higher performance in precision than recall since the entity list may not include all types of entities. For example, location names include a wide range, from city names, region names, and area names to mountain names, temple names, and station names. However, not all of these entities are included in the open government data.

In contrast, our system gives a slightly lower F-measure on the Japanese and English business organization name recognition via tri-training. As Japanese language contains kanji (日文漢字, かんじ), hiragana (平仮名, ひらがな), and katakana (片仮名, カタカナ), frequent pattern mining may generate dictionaries that contain incomplete keywords for features. Meanwhile, we currently use a fixed number (100) of vocabularies for each dictionary, which may not accommodate enough patterns for Japanese and English.

5. CONCLUSION

Named entity extraction has been approached with supervised approaches that require large labeled training examples to achieve good performance. This research makes use of automatic labeling based on known entity names to create a large corpus of labeled training data. While such data may contain noise, the benefit with large labeled training data still is more significant than the noise it inherits. In practice, we might have a large amount of unlabeled data. Therefore, we applied tri-training to make use of such unlabeled data and to modify the colabeling mechanism for sequence labeling to improve the performance. Instead of assuming a constant error rate for the initial error of each classifier, we proposed a new way to estimate the number of examples selected from unlabeled data. In this article, we started by building a web NER model for Chinese person name extraction by constructing the dictionaries manually and then extended the framework to four different NER tasks by constructing the dictionaries automatically via frequent pattern mining. As shown in the experiments, the proposed approach outperforms the state-of-the-art Stanford NER tool for three Chinese NER tasks.

APPENDIX

Tri-Training Pseudo Code

ALGORITHM 1: Tri-Training4SeqLabeling

Input: L:Original Labeled example set
U:Unlabeled example set
Learn: Learn Algorithm
$i, j, k \in \{1, 2, 3\}, i \neq j \neq k$
Output: $h(x) \leftarrow CoLabeling\_Testing(x, h_i, h_j, h_k),\ x \in T$ `// ref. Equation (14)`

for $i \in \{1 \dots 3\}$ **do**
  $BS_i \leftarrow BootstrapSample(L)$, $h_i \leftarrow Learn(BS_i)$, $t \leftarrow 1$
end
repeat
  for $i \in \{1 \dots 3\}$ **do**
    $S_i^t \leftarrow \phi$
    $e_i^t \leftarrow MeasureError(h_j, h_k)\ (j, k \neq i)$ `// ref. Equation (2)`
    $update_i \leftarrow False$

    `// Estimate the upper bound of new examples`
    if $t == 1$ **then**
      $u \leftarrow \left\lceil \frac{(|L_i^W(h_j,h_k)|+1)|L^C(h_j,h_k)|}{|L^W(h_j,h_k)|+1} - 1 \right\rceil$ `// ref. Equation (10)`
      $u \leftarrow \min(|L|, u)$ `// ref. Equation (11)`
      $update_i \leftarrow True$
    else if $e_i^t < e_i^{t-1}$ **then**
      $u \leftarrow \left\lceil \frac{(e_i^{t-1}+\epsilon)|S_i^{t-1}|}{e_i^t+\epsilon} - 1 \right\rceil$ `// ref. Equation (3)`
      $u \leftarrow \min(|L| + |S_i^{t-1}|, u)$ `// ref. Equation (11)`
      $update_i \leftarrow True$
    else
      $u \leftarrow 0$ `// update` $|S_i^t|$ `for next round`
      $|S_i^t| \leftarrow |S_i^{t-1}|$
    end

    `// Ramdon pick new examples from U` ($|U| \gg |S_i^t|$)
    if $update_i == True$ **then**
      while $|S_i^t| < u$ **do**
        pick x from U
        $y \leftarrow CoLabeling\_Training(x, h_j, h_k)$ `// ref. Equation (13)`
        if $y \neq Null$ **then**
          $S_i^t \leftarrow S_i^t \cup (x, y)$
        end
      end
    end
  end
  for $i \in \{1 \dots 3\}$ **do**
    if $update_i == True$ **then**
      $h_i \leftarrow Learn(L \cup S_i^t)$
    end
  end
  $t \leftarrow t + 1$
until $update_i == False, i \in \{1 \dots 3\}$

Table VII. Tri-Training Log on DS1

Round	Classifier	$Error\,Rate^t$	u	$\lvert L\rvert + \lvert S^{t-1}\rvert$	Retrain?
1	h_1	$\frac{79}{4,594} = 0.0172$	2,384	4,637	True
	h_2	$\frac{70}{4,591} = 0.0152$	2,755	4,637	True
	h_3	$\frac{64}{4,593} = 0.0139$	2,584	4,637	True
2	h_1	$\frac{16}{4,635} = 0.0035$	11,876	7,021	True
	h_2	$\frac{11}{4,636} = 0.0024$	17,703	7,392	True
	h_3	$\frac{14}{4,635} = 0.0030$	11,920	7,221	True
3	h_1	$\frac{13}{4,635} = 0.0028$	8,641	11,658	True
	h_2	$\frac{12}{4,636} = 0.0026$	0	12,029	False
	h_3	$\frac{14}{4,636} = 0.0030$	7,222	11,858	True
4	h_1	$\frac{12}{4,634} = 0.0026$	9,359	13,278	True
	h_2	$\frac{9}{4,637} = 0.0019$	9,036	12,029	True
	h_3	$\frac{12}{4,631} = 0.0026$	8,416	11,859	True
5	h_1	$\frac{14}{4,636} = 0.0030$	0	13,996	False
	h_2	$\frac{12}{4,635} = 0.0026$	0	13,673	False
	h_3	$\frac{13}{4,635} = 0.0028$	0	13,053	False

Tri-Training Log

To demonstrate the idea of the proposed tri-training algorithm, we use Table VII to illustrate the number of training examples used in each iteration. For the first round, we set u by Equation (10) (line 11) and limit the value by $|L|$ (Equation (11), line 12) to avoid the domination of newly added examples. At each iteration, we calculate the error rate by Equation (2) (line 7) and estimate the amount of new examples u via Equation (3) (line 15). The difference is that u is now bounded by the amount of previous training examples, $|L| + |S^{t-1}|$ (line 16). For this example with 4,637 training examples in L, three classifiers are retrained with more training examples whenever the errors are decreased. If the error rate is increased, for example, h_2 at the third round, the classifier will not be retrained. This upper bound provides a mechanism to control the behaviors of the three classifiers with good performance for each round. Finally, the tri-training algorithm stops at the fifth round when all error rates increase.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their constructive comments, especially for the suggestions on the experimentation details and comparison with other existing NER systems.

REFERENCES

Joohui An, Seungwoo Lee, and Gary Geunbae Lee. 2003. Automatic acquisition of named entity tagged corpus from world wide web. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 2 (ACL'03)*. Association for Computational Linguistics, Stroudsburg, PA, 165–168. DOI:http://dx.doi.org/10.3115/1075178.1075207

Rie Kubota Ando and Tong Zhang. 2005. A high-performance semi-supervised learning method for text chunking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics (ACL'05)*. Association for Computational Linguistics, Stroudsburg, PA, 1–9. DOI:http://dx.doi.org/10.3115/1219840.1219841

Kristin P. Bennett and Ayhan Demiriz. 1999. Semi-supervised support vector machines. In *Proceedings of the 1998 Conference on Advances in Neural Information Processing Systems II*. MIT Press, Cambridge, MA, 368–374. http://dl.acm.org/citation.cfm?id=340534.340671

Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT'98)*. ACM, New York, NY, 92–100. DOI:http://dx.doi.org/10.1145/279943.279962

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM*

SIGMOD International Conference on Management of Data (SIGMOD'08). ACM, New York, NY, 1247–1250. DOI:http://dx.doi.org/10.1145/1376616.1376746

Andrew Eliot Borthwick. 1999. *A Maximum Entropy Approach to Named Entity Recognition*. Ph.D. Dissertation. New York, NY. Advisor(s) Grishman, Ralph. AAI9945252.

John Burger, Claire Cardie, Vinay Chaudhri, Robert Gaizauskas, Sanda Harabagiu, David Israel, Christian Jacquemin, Chin-Yew Lin, Steve Maiorano, George Miller, Dan Moldovan, Bill Ogden, John Prager, Ellen Riloff, Amit Singhal, Rohini Shrihari, Tomek Strzalkowski, Ellen Voorhees, and Ralph Weischedel. 2001. Issues, Tasks and Program Structures to Roadmap Research in Question & Answering (Q&A). Technical Report. NIST. http://www-nlpir.nist.gov/projects/duc/roadmapping.html.

Erik Cambria, Bjorn Schuller, Yunqing Xia, and Catherine Havasi. 2013. New avenues in opinion mining and sentiment analysis. *IEEE Intelligent Systems* 28, 2 (March 2013), 15–21. DOI:http://dx.doi.org/10.1109/MIS.2013.30

Chia-Hui Chang and Shu-Ying Li. 2010. MapMarker: Extraction of postal addresses and associated information for general web pages. In *Web Intelligence*, Jimmy Xiangji Huang, Irwin King, Vijay V. Raghavan, and Stefan Rueger (Eds.). IEEE Computer Society, 105–111.

Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien. 2006. *Semi-Supervised Learning*. http://www.amazon.com/Semi-Supervised-Learning-Author-Chapelle-Oct-2006/dp/B010DTUKDY/ref=sr_1_4?s=books&ie=UTF8&qid=1462116256&sr=1-4

Wenliang Chen, Yujie Zhang, and Hitoshi Isahara. 2006. Chinese chunking with tri-training learning. In *Proceedings of the 21st International Conference on Computer Processing of Oriental Languages: Beyond the Orient: The Research Challenges Ahead (ICCPOL'06)*. Springer-Verlag, Berlin, 466–473. DOI:http://dx.doi.org/10.1007/11940098_49

Hsiu-Min Chuang, Chia-Hui Chang, and Ting-Yao Kao. 2014. Effective web crawling for chinese addresses and associated information. In *E-Commerce and Web Technologies*. Springer, 13–25.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics (ACL'05)*. Association for Computational Linguistics, Stroudsburg, PA, 363–370. DOI:http://dx.doi.org/10.3115/1219840.1219885

Ruiji Fu, Bing Qin, and Ting Liu. 2011. Generating Chinese named entity data from a parallel corpus. In *Proceedings of 5th International Joint Conference on Natural Language Processing*. 264–272.

Abhishek Gattani, Digvijay S. Lamba, Nikesh Garera, Mitul Tiwari, Xiaoyong Chai, Sanjib Das, Sri Subramaniam, Anand Rajaraman, Venky Harinarayan, and AnHai Doan. 2013. Entity extraction, linking, classification, and tagging for social media: A wikipedia-based approach. *Proceedings of the VLDB Endowment* 6, 11 (Aug. 2013), 1126–1137. DOI:http://dx.doi.org/10.14778/2536222.2536237

Sally A. Goldman and Yan Zhou. 2000. Enhancing supervised learning with unlabeled data. In *Proceedings of the 17th International Conference on Machine Learning (ICML'00)*. Morgan Kaufmann Publishers, San Francisco, CA, 327–334. http://dl.acm.org/citation.cfm?id=645529.658273

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'04)*. ACM, New York, NY, 168–177. DOI:http://dx.doi.org/10.1145/1014052.1014073

Taku Kudo. 2005. CRF++: Yet Another CRF toolkit. (2005). http://crfpp.googlecode.com.

Jimmy Lin. 2002. The web as a resource for question answering: Perspectives and challenges. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC'02)*.

Gideon S. Mann and Andrew McCallum. 2010. Generalized expectation criteria for semi-supervised learning with weakly labeled data. *Journal of Machine Learning Research* 11 (March 2010), 955–984. http://dl.acm.org/citation.cfm?id=1756006.1756038

Andrew McCallum, Dayne Freitag, and Fernando C. N. Pereira. 2000. Maximum entropy Markov models for information extraction and segmentation. In *Proceedings of the 17th International Conference on Machine Learning (ICML'00)*. Morgan Kaufmann Publishers, San Francisco, CA, 591–598. http://dl.acm.org/citation.cfm?id=645529.658277

Andrew McCallum and Wei Li. 2003. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the 7th Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4 (CONLL'03)*. Association for Computational Linguistics, Stroudsburg, PA, 188–191. DOI:http://dx.doi.org/10.3115/1119176.1119206

Matthew Michelson and Craig A. Knoblock. 2009. Exploiting background knowledge to build reference sets for information extraction. In *Proceedings of the 21st International Joint Conference on Artifical Intelligence (IJCAI'09)*. Morgan Kaufmann Publishers, San Francisco, CA, 2076–2082. http://dl.acm.org/citation.cfm?id=1661445.1661777

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 (ACL'09)*. Association for Computational Linguistics, Stroudsburg, PA, 1003–1011. http://dl.acm.org/citation.cfm?id=1690219.1690287

Tri Thanh Nguyen, Le Minh Nguyen, and Akira Shimazu. 2008. Using semi-supervised learning for question classification. *Information and Media Technologies* 3, 1 (2008), 112–130. DOI:http://dx.doi.org/10.11185/imt.3.112

Kamal Nigam and Rayid Ghani. 2000. Analyzing the effectiveness and applicability of co-training. In *Proceedings of the 9th International Conference on Information and Knowledge Management (CIKM'00)*. ACM, New York, NY, 86–93. DOI:http://dx.doi.org/10.1145/354756.354805

Xipeng Qiu, Qi Zhang, and Xuanjing Huang. 2013. FudanNLP: A toolkit for Chinese natural language processing. In *Proceedings of Annual Meeting of the Association for Computational Linguistics*.

Adam Rae, Vanessa Murdock, Adrian Popescu, and Hugues Bouchard. 2012. Mining the web for points of interest. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'12)*. ACM, New York, NY, 711–720. DOI:http://dx.doi.org/10.1145/2348283.2348379

Ellen Riloff, Janyce Wiebe, and Theresa Wilson. 2003. Learning subjective nouns using extraction pattern bootstrapping. In *Proceedings of the 7th Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4 (CONLL'03)*. Association for Computational Linguistics, Stroudsburg, PA, 25–32. DOI:http://dx.doi.org/10.3115/1119176.1119180

Sunita Sarawagi. 2008. Information extraction. *Foundational Trends Databases* 1, 3 (March 2008), 261–377. DOI:http://dx.doi.org/10.1561/1900000003

L. Satish and B. I. Gururaj. 1993. Use of hidden Markov models for partial discharge pattern classification. *IEEE Transactions on Electrical Insulation,* 28, 2 (Apr. 1993), 172–182. DOI:http://dx.doi.org/10.1109/14.212242

Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2005. Learning syntactic patterns for automatic hypernym discovery. In *Advances in Neural Information Processing Systems 17*, L. K. Saul, Y. Weiss, and L. Bottou (Eds.). MIT Press, 1297–1304. http://papers.nips.cc/paper/2659-learning-syntactic-patterns-for-automatic-hypernym-discovery.pdf.

Yueng-Sheng Su. 2012. Associated Information Extraction for Enabling Entity Search on Electronic Map. (2012). http://ir.lib.ncu.edu.tw:88/thesis/view_etd.asp?URN=955202022.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the 7th Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4 (CONLL'03)*. Association for Computational Linguistics, Stroudsburg, PA, 142–147. DOI:http://dx.doi.org/10.3115/1119176.1119195

L. G. Valiant. 1984. A theory of the learnable. *Communications of the ACM* 27, 11 (Nov. 1984), 1134–1142. DOI:http://dx.doi.org/10.1145/1968.1972

Konstantinos N. Vavliakis, Andreas L. Symeonidis, and Pericles A. Mitkas. 2013. Event identification in web social media through named entity recognition and topic modeling. *Data Knowledge Engineering* 88 (Nov. 2013), 1–24. DOI:http://dx.doi.org/10.1016/j.datak.2013.08.006

V. A. Yatsko, M. S. Starikov, and A. V. Butakov. 2010. Automatic genre recognition and adaptive text summarization. *Automatic Documentation and Mathematical Linguistics* 44, 3 (June 2010), 111–120. DOI:http://dx.doi.org/10.3103/S0005105510030027

Ning Yu and Sandra Kubler. 2010. Semi-supervised learning for opinion detection. In *2014 IEEE / WIC / ACM International Joint Conferences on Web Intelligence (WI'14) and Intelligent Agent Technologies (IAT'14) - Volume* 3, 249–252. DOI:http://dx.doi.org/10.1109/WI-IAT.2010.263

Zhi-Hua Zhou and Ming Li. 2005. Tri-training: Exploiting unlabeled data using three classifiers. *IEEE Transactions on Knowledge and Data Engineering* 17, 11 (Nov. 2005), 1529–1541. DOI:http://dx.doi.org/10.1109/TKDE.2005.186

Received October 2015; revised May 2016; accepted June 2016